Queensland University of Technology
Brisbane Australia

This is the author's version of a work that was submitted/accepted for publication in the following source:

Gluga, Richard, Kay, Judy, Lister, Raymond, Simon, Simply, Charleston, Michael, Harland, James, & Teague, Donna M. (2013) A conceptual model for reflecting on expected learning vs. demonstrated student performance. In Carbone, Angela & Whalley, Jacqueline (Eds.) *Proceedings of the 15th Australasian Computing Education Conference (ACE2013)*, Australian Computer Society, Inc., Adelaide, SA.

This file was downloaded from: http://eprints.qut.edu.au/57673/

© Copyright 2013 Australian Computer Society, Inc.

Copyright c 2013, Australian Computer Society, Inc. This pa- per appeared at the 15th Australasian Computer Education Conference (ACE 2013), Adelaide, South Australia, January- February 2013. Conferences in Research and Practice in Infor- mation Technology (CRPIT), Vol. 136, Angela Carbone and Jacqueline Whalley, Ed. Reproduction for academic, not-for- profit purposes permitted provided this text is included.

Notice: *Changes introduced as a result of publishing processes such as copy-editing and formatting may not be reflected in this document. For a definitive version of this work, please refer to the published source:*

A conceptual model for reflecting on expected learning vs. demonstrated student performance

Richard Gluga [1] **Judy Kay** [1] **Raymond Lister** [2] **Simon**[3]
Michael Charleston[1] **James Harland**[4] **Donna Teague**[5]

[1] School of IT, University of Sydney, Sydney NSW Australia
[2] School of Software, University of Technology Sydney, Sydney NSW Australia
[3] School of Design Communication and IT, University of Newcastle, Newcastle NSW Australia
[4] School of Computer Science and Information Technology, RMIT University, Melbourne VIC Australia
[5] Faculty of Science and Technology, Queensland University of Technology, Brisbane QLD Australia

Abstract

Educators are faced with many challenging questions in designing an effective curriculum. What prerequisite knowledge do students have before commencing a new subject? At what level of mastery? What is the spread of capabilities between bare-passing students vs. the top-performing group? How does the intended learning specification compare to student performance at the end of a subject? In this paper we present a conceptual model that helps in answering some of these questions. It has the following main capabilities: capturing the learning specification in terms of syllabus topics and outcomes; capturing mastery levels to model progression; capturing the minimal vs. aspirational learning design; capturing confidence and reliability metrics for each of these mappings; and finally, comparing and reflecting on the learning specification against actual student performance. We present a web-based implementation of the model, and validate it by mapping the final exams from four programming subjects against the ACM/IEEE CS2013 topics and outcomes, using Bloom's Taxonomy as the mastery scale. We then import the itemised exam grades from 632 students across the four subjects and compare the demonstrated student performance against the expected learning for each of these. Key contributions of this work are the validated conceptual model for capturing and comparing expected learning vs. demonstrated performance, and a web-based implementation of this model, which is made freely available online as a community resource.

Keywords: curriculum, assessment, course content

1 Introduction

To develop an effective teaching and learning plan for a subject that has prerequisites, a lecturer must be aware of the capabilities of the students at the beginning of that subject. That is, the lecturer must have a solid idea of the knowledge and concepts that students have learnt in the previous semester, and the level of mastery achieved. The teaching schedule, lecture topics and learning outcome statements from the previous subject may provide some indication as to the content that was covered, but this does not detail what was actually assessed, how it was assessed and how it was graded. The marking criteria for the subject might have awarded most marks for rote-memorisation of algorithms and code recipes. On the other hand, perhaps the assessments tested higher-level problem-solving skills using the learnt concepts in unfamiliar scenarios. It is not easy to discern how much of the overall assessment weight was associated to the former as opposed to the latter. The lecturer, however, must be aware of these details in order to develop an effective teaching program based on the capabilities of beginning students.

Likewise, a lecturer must be able to answer the same questions about the teaching and assessments of his or her own subject. That is, which topics and concepts are expected as subject outcomes, and at what levels of mastery are students expected to achieve them? Additionally, what does the assessment design infer or guarantee about the minimal capabilities expected of bare-passing students at the end of the subject, and how does this compare to the aspirational outcomes expected of top-performing students?

Further still, expected outcomes must be validated against actual learning, as demonstrated by student performance, to ensure that the teaching and learning design is realistic. That is, are bare-passing students meeting the minimal expectations? Are top-performing students achieving the aspirational outcomes? If expectations do not align with demonstrated performance, the lecturer must consider why, and what remedial teaching or assessment changes are appropriate for future offerings of the subject.

Taking a whole program perspective, each individual subject is only one in a long sequence of 24 or more in a typical three- or four-year degree. From semester to semester, students must progressively learn new concepts and build upon the concepts previously learnt. So in order to develop an effective program sequence, each subject lecturer must be able to answer these questions about his or her own subject, and about previous subjects in the sequence. The many subject lecturers involved in the teaching of a degree program must thus have a shared and comparable understanding of the outcomes and mastery levels developed throughout the program.

This paper presents a conceptual model for a systematic curriculum mapping and learner modelling approach that enables subject lecturers to design and document the learning goals in a subject, and to compare expected learning with actual performance as demonstrated by student assessment grades. This is done in terms of a syllabus specification that can be used to communicate learning goals across a whole computer science degree program. The conceptual model also formally captures the level of mastery for each topic or outcome assessed, and the academic's

Copyright ©2013, Australian Computer Society, Inc. This paper appeared at the 15th Australasian Computer Education Conference (ACE 2013), Adelaide, South Australia, January-February 2013. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 136, Angela Carbone and Jacqueline Whalley, Ed. Reproduction for academic, not-for-profit purposes permitted provided this text is included.

confidence and judgement of reliability for each of these. This model enables academics to systematically answer some of the difficult questions posed above.

2 Background

Sadler (2009) claims that "academic achievement standards is now the key issue. It is what worries a lot of people". He asks "do the grades that are on students' transcripts actually mean what they say?" That is, what do assessment marks actually tell us, if we cannot reliably identify what exactly is being assessed, what cognitive skills are required to pass the assessments, and what a bare-passing grade means compared to the highest passing grade. Sadler suggests that "what we need to do is find ways of capturing the standards we want to use, so we can compare students' work with those standards". Doing so means that "each grade represents a particular level of competence, knowledge or skill", and as Sadler put it, "that is the crux of the matter".

Similar concerns have also been expressed in the computing education (CSEd) research community. Commenting on the Grand Challenges facing computing education, McGettrick (2005) makes several points about important issues relating to the computer science curriculum, including that there is an increasing need for curriculum standardisation and for comparable outcomes. This is due to the continuing globalisation of the workforce, which requires students, educators, employees and employers to have a common vocabulary for describing discipline skills and competence levels. McGettrick observes that "there are different levels of learning as exhibited by the existence of Bloom's taxonomy of educational objectives (Bloom et al. 1956). These different levels, as well as the associated degrees of commitment required to achieve these levels, need recognition and their consequences understood". Two of the grand challenges which relate directly to this are:

- Identify very clearly the technical skills ... that students should acquire throughout their program of study in higher education [2.3.2.i]
- Identify and then employ a phased development of all these skills, ensuring that the skill levels are such that graduates are internationally competitive in terms of their skills... [2.3.2.ii]

2.1 Learning Standards in Computer Science

In order to implement the transparency proposed in the previous section, there needs to be an agreed set of learning goals against which to measure student performance. For computer science disciplines within Australia there are several candidate sets of learning goals that might be useful. These include high-level transferable generic graduate attributes (Barrie et al. 2009), national graduate outcomes such as the upcoming TEQSA TLOs (ALTC 2010), international standards such as the ABET-CAC accreditation guidelines (ABET 2011), and fine-grained Syllabus or Body of Knowledge topic and outcome recommendations such as those from the ACS (Gregor et al. 2008) or the ACM and IEEE (ACM/IEEE 2008, 2013).

In this paper we choose to focus on detailed fine-grained syllabus outcomes, and specifically those from the CS2013 Strawman (ACM/IEEE 2013), which lists over 1366 topics and 1041 learning objectives, categorised into 18 top-level Knowledge Areas and 155 Knowledge Units. Out of the 1366 topics, 257 are classified as Tier-1 Core (absolute essentials), 328 as Tier-2 Core (80% minimum coverage expected) and 781 as electives. Whilst Australian computer science degree programs are not formally accredited against this curriculum, most institutions endeavour to be mindful of and align with these recommendations. Additionally, the ACM/IEEE CS guideline is one of the most comprehensive and widespread Body of Knowledge descriptions of a computer science degree. As such, it provides a common vocabulary for describing and sharing the design of teaching, learning and assessment activities, both among the different subject lecturers within an institution and across institutions within the wider computer science discipline.

2.2 Mastery and Progression in Computer Science

As well as indicating the need for agreed learning standards, both Sadler and McGettrick proposed that students' level of competence or mastery would need to be measured against such learning standards. Much research has been published about the importance of this in the CSEd community. Lister & Leaney (2003*a*,*b*) proposed a criterion-based grading scheme based on Bloom's Taxonomy (Bloom et al. 1956), where bare-passing students are expected to show competence at the novice levels (Knowledge and Comprehension) while top-performing students should be challenged at the higher levels (Synthesis and Evaluation). Similar uses of Bloom's Taxonomy to classify the cognitive complexity of programming exercises have been discussed by many others (Reynolds & Fox 1996, Buck & Stucki 2001, Oliver et al. 2004, Burgess 2005, Whalley et al. 2006, Starr et al. 2008, Thompson et al. 2008, Gluga, Kay, Lister, Kleitman & Lever 2012, Simon et al. 2012). Bloom's Taxonomy is also the recommended medium for specifying mastery in the CS2008 curriculum (ACM/IEEE 2008) and in the ACS ICT Profession Body of Knowledge (Gregor et al. 2008). The new ACM/IEEE CS2013 Strawman has made a slight departure from Bloom's Taxonomy, proposing instead a new three-level mastery scale, the merits of which are currently under review (Lister 2012).

2.3 Curriculum Mapping

Having found suitable learning standards (the CS2013 Strawman) and a suitable cognitive classification theory (Bloom's Taxonomy) on which to model our computer science degree programs, we then turned to literature on curriculum mapping as a framework on which to construct our model. English (1988) proposed that an effective approach to curriculum management "should include a planned relationship between the written, taught and tested curricula". English stated that effective program planning and auditing "should ensure that the written curriculum has planned relationships to the taught curriculum, and that the taught curriculum and written curriculum are related to the tested curriculum".

English (1978) also stated that "curriculum guidelines, behavioral objectives, course outlines are all descriptions of a future desired condition" and thus "do not represent the actual curriculum applied by individual teachers". He saw this as a serious problem, labeling curriculum guides and course outlines as the *fictional* curriculum. He stated that "to exercise quality control over curriculum requires the instructional leader or supervisor to know what the *real* curriculum is in his or her subject area" and unless the real curriculum is "known and quantified, it is not possible to understand ... existing gaps or holes" in the program of study. English proposed that "a fairly accurate picture of the real curriculum" must be obtained in order to allow for effective quality control.

Curriculum mapping has been used extensively in K-12 education in the United States (Jacobs 1989, 1991, 1997, 2010). In tertiary education, however, it has been adopted mostly by the medical disciplines

(Willett 2008, Britton et al. 2008, Harden 2001), and more recently to some extent by engineering (Gluga et al. 2010, Wigal 2005) and other professionally accredited disciplines. Examples of such systems in computer science education are limited. One example is the COMPASS system, developed as a Moodle plugin at the University of West Georgia (Abunawass et al. 2004); COMPASS provided mechanisms to link the assessment in each subject to CC2001 topics and learning objectives, at appropriate Bloom mastery levels. This system aimed to answer some of the same questions we identified earlier.

However, COMPASS had a number of limitations. Data entry was "a bit daunting" (Abunawass et al. 2004), in that users had to open external websites to read through syllabus specifications and manually copy over the appropriate topics/outcomes for each assessment mapping. Additionally, "most administrative and review functions require direct interaction with the underlying database using SQL commands", which meant that visualising the mapped relationships required significant technical expertise and manual data processing. Further still, the system did not integrate with student marks (this was listed as future work, but no further related publications could be found), so there was no way to compare the actual student performance with the intended curriculum design.

3 Conceptual Model

To enable subject lecturers to plan more effective and integrative teaching and learning activities, we have developed a conceptual model for documenting and describing degree programs in terms of well defined learning goals and mastery levels. The conceptual model supports the capture of teaching and learning intention at multiple curriculum stages, based on the ideas introduced by English and others. This model is represented in Figure 1. We define five curriculum stages as follows (leftmost column in the Figure).

- *Recommended Curriculum* – the collection of graduate attributes, national/international learning standards, accreditation competencies and syllabus or body of knowledge recommendations that are relevant for each degree program. A degree program may not need to consider all recommendations, but may aspire to do so for accreditation purposes and recognition purposes. In this paper we focus on fine-grained discipline specific topics and outcomes from an authoritative syllabus, namely the ACM/IEEE Computer Science Curriculum Guidelines (CS2013) (ACM/IEEE 2013).
- *Planned Curriculum* – the structure of a typical three- to five-year degree program, comprising two semesters per year and four core (C) or elective (E) subjects per semester. Each core and elective subject must contribute towards the learning goals from the Recommended Curriculum that the degree program aspires to align with, such as the 1366 topics and 1041 outcomes of the CS2013. Significant planning is required to decide which topics are to be covered in which subjects, and at which levels of mastery, to ensure an effective progressive sequence of study.
- *Practised Curriculum* – the outcomes and learning activities in every subject. The outcomes for the subject are the lecturer's interpretation of the aims of the subject, based on the learning goals prescribed as part of the program-level Planned Curriculum. These outcomes thus drive the prerequisite knowledge of the subject, and also the design of learning activities such as lecture topics, lab exercises, text readings, etc.
- *Assessed Curriculum* – the learning goals that are actually assessed as part of each individual subject. The Assessed Curriculum is defined by the subject lecturer when creating the assessment exercises for the class. Each assessment question or task may relate to one or more recommended, planned and practised learning goals.
- *Demonstrated Curriculum* – a description of what students have actually learnt as part of a subject or collection of subjects, based on the fine-grained marks associated with each assessment exercise. The Demonstrated Curriculum is a profile of learners in terms of learning goals and mastery levels achieved, based on the marks from each assessed component.

In this paper we focus on the effectiveness of this conceptual model in enabling subject lecturers to describe the Assessed Curriculum and the Demonstrated Curriculum in terms of fine-grained syllabus/body-of-knowledge learning goals and mastery levels. The model supports description and comparison of the expected performance of the bare-passing student vs. the top-performing student vs. demonstrated student performance in terms of these learning goals and mastery levels. The model additionally supports a mechanism for capturing the academic's confidence as to the reliability of each classification, such that a confidence value may be used to express overall certainty or uncertainty in each of the presented visualisations. These aspects are discussed in greater detail in the following subsections.

3.1 Modelling the Assessed Curriculum

The Assessed Curriculum represents the subject lecturer's expectations as to what bare-passing students and top-performing students will have learnt, and will be able to demonstrate, at the end of the subject. This is represented in the Assessed Curriculum section of Figure 1 as a collection of exams or assessments designed to measure student learning. Each exam or assessment is broken down into a set of questions or sub-tasks, which are graded separately and may assess different learning goals, at different levels of mastery.

Subjects, exams and questions each have a weight component as a function of performance in the overall degree program. That is, a subject usually has a credit-point value, an exam or assessment has an overall subject weight, and a question or task is worth a certain number of marks. These are important for capturing and calculating the strength of evidence for each modelled learning goal and mastery level, as will be discussed later.

On the right side of the Assessed Curriculum box in Figure 1, we show how learning goals, mastery levels, and other elements are mapped to each assessment question. We label these mappings the *Academic Classifications*. The first of these is a *reliability* score that is associated with each assessment. This score represents the academic's judgement of how reliable the grades from the classified exam or assessment are considered to be. For example, an academic may feel that an end-of-semester closed-book written final exam, completed under strict supervision, is a fairly accurate representation of a student's capabilities. On the other hand, a take-home assessment may be considered less reliable as an indicator, as the student is easily able to seek external help in completing it, and thus the final mark may not be as reliable an indicator of the student's actual capabilities.

The remaining four fields in the Academic Classification box map to each assessment question. The first is a *bare-pass friendly* yes/no flag, which indicates if

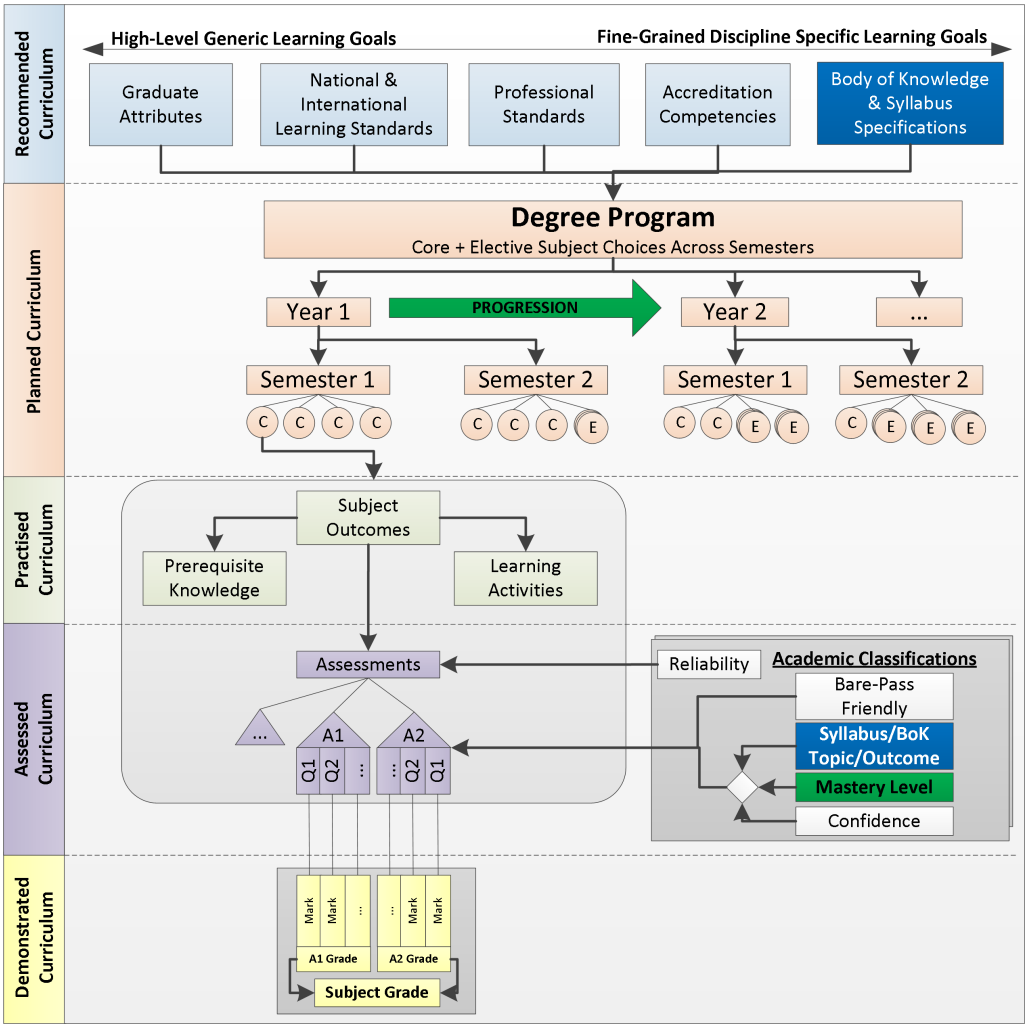

Figure 1: Conceptual model for degree program curriculum design

the academic expects most of the students who finish the subject with a bare-pass mark to be able to earn, say, 70% or more of the marks in that question.

The remaining three mappings (*topic/outcome, mastery level, confidence*), are stored together as a single sub-classification, and are defined as follows:

1. *Topic/outcome* – a mapping to a relevant topic, outcome or other learning goal from the Recommended Curriculum which is specifically assessed by the question being classified.
2. *Mastery level* – a classification of the cognitive difficulty at which the mapped topic or outcome is assessed (that is, a student would be expected to be operating at this minimal level to answer the question correctly)
3. *Confidence* – a score from 0 to 100 representing the academic's confidence in the validity of this classification.

Multiple instances of such sub-classifications can be made for each question: a question may assess multiple learning goals, each at different levels of mastery (for example, a question may require an advanced understanding of the topic *loops and iteration* but only basic familiarity with *arrays*). The confidence meta-tag can be used to represent any uncertainty in each sub-classification. In some instances it may not be easy to define the mastery level at which a specific topic or objective is being assessed, in which case a low confidence rating can be specified. In other circumstances the academic may feel that even though a topic is being assessed, it is only a small part of the overall question, so again a low confidence rating may be used to record this as a low evidence mapping.

3.2 Modelling the Demonstrated Curriculum

The Demonstrated Curriculum represents the learning goals and mastery demonstrated by students at the completion of a subject, a set of subjects, or a whole degree program. This is achieved by collecting itemised student marks for each question or task, and using these to compute the achieved level of mastery across the subject/s or program as a whole, or for specific topics or outcomes. These performance scores may then be compared against the Assessed Curriculum design to see how closely they match the modelled expectations.

3.3 Algorithm for Aggregating Classifications

The algorithm for aggregating the assessment question classification data into meaningful forms is as follows.

(i) Calculate the weight of each question (Q_w) as a proportion of the overall subject weight (S) and of the overall degree program (P). Let the credit-point value of the subject be S_{cp}, the weight of each assessment be A_w, and the marks for each question be Q_m, giving $Q_w = (S_{cp}/P_{cp}) \cdot A_w \cdot (Q_m/A_m)$.

(ii) Next, inspect the topic/outcome mappings for each question (TOM). The evidence score for each topic/outcome mapping (TOM_e) is given by the weight of the question (Q_w) divided by the number of topic/outcome mappings for that question (Q_{nTOM}), multiplied by its confidence rating (TOM_{conf}) and the exam/assessment reliability rating (A_{rel}) to give

the final $TOM_e = Q_w \cdot TOM_{conf} \cdot A_{rel}/Q_{nTOM}$.

This now gives a list of evidence scores for each assessed topic/outcome mapping. The sum of all evidence scores for a specific topic or outcome can be used to represent the overall assessment weight associated with that topic across the whole subject, or across the whole degree program. Additionally, each topic/outcome mapping also has a mastery level classification, so we can also sum up all topic/outcome mappings at a specific mastery level to represent the overall assessment weight associated with that level. A third possibility is to sum up all the TOM_e scores for a specific topic/outcome at a specific mastery level.

Further still, we can separate the TOM_e scores into two categories: those from questions that were marked as bare-pass friendly and those from questions intended to distinguish top-performing students. This allows us to create models of the assessed curriculum showing the expected performance of bare-passing students (the minimal standard) vs. the expected performance of top-performing students (the aspirational standard) in relation to the mapped syllabus and mastery levels.

To compute and generate the demonstrated curriculum models, we simply factor the average mark across a set of students for a specific question and multiply this by the topic/outcome evidence score from above. This enables a side-by-side comparison of the expected outcomes of bare-pass students vs. expected outcomes of top-performing students, and vs. actual outcomes of any group of students.

4 User View

We have implemented the conceptual model described above as part of our ProGoSs research system, which aims to enable educators to document the learning across a whole computer science degree and represent it in terms of authoritative curriculum specification. The research presented here is one aspect of the broader BABELnot project (Lister et al. 2012), which aims to document and benchmark the academic standards associated with the core sequence of programming subjects in computer science degrees.

The ProGoSs system allows users to specify the core and elective subjects of a degree program, and then, for each subject, a list of assessments or exams and a sub-list of questions or tasks.

Figure 2 shows the interface for classifying Question 8 from a fictitious final exam in a first semester programming fundamentals subject. The system allows the user to write or copy-paste the actual question text, or to upload an image, or to leave the text empty and instead reference a PDF version of the exam. The question shown in the figure is worth 5 marks, and requires students to *Write a function to return the minimum integer in an array.* Below the question text is the classification meta-data, including the *bare-pass friendly* flag as discussed earlier, and two additional meta-fields for familiarity and estimated time required for students to answer the question. These two fields are not currently used in any further processing.

The lower half of Figure 2 shows the topic/outcome, mastery level, and confidence classifications. In this case, the question was mapped to three topics from the ACM/IEEE CS2013. The large slider on the right is used to quickly set the mastery level for each topic, in this case using Bloom's Taxonomy. All three topics have been mapped at the Application level. Moving the slider left or right moves through the six Bloom levels, with Knowledge to the far left and Evaluation to the far right.

Beneath the mastery sliders is a smaller slider, which can be used to record the user's confidence in each mapping decision. In this case, all three confidence sliders are set to 100%, indicating the user is very confident in the mappings made. This process is repeated to map all of the questions in a particular exam or assessment.

Additional topic or outcome mappings can be added through a floating dialog editor which allows the user to begin typing a keyword, such as 'parameters', whereupon any matching topics or outcomes from the linked syllabus document will be instantly displayed on the screen. From here, the user can use the sliders to immediately assign a mastery level and confidence, and continue searching for other keywords. The dialog additionally supports manual browsing through the syllabus hierarchy of knowledge areas and knowledge units to select relevant topics. A user may also define his or her own set of topics or outcomes, which may be used in combination with, or instead of, an authoritative curriculum.

The tabs across the top of Figure 2 allow the user to access a range of other functionalities, namely:

- *Overview* – brief description of subject details, typically similar to what appears in a printed handbook.
- *Prerequisites* – mapping of syllabus topics/outcomes and mastery levels that represents expected student knowledge prior to commencing the subject.
- *Assessments* – list of subject exams and assessments, including facility to drill down to individual questions as seen in Figure 2.
- *Dependency checks* – compares the specified prerequisite topics/outcomes to previous subjects in the degree program sequence, allowing the subject lecturer to quickly identify where, and to what extent, each prerequisite topic/outcome was taught and assessed. Similarly, this screen also shows subsequent subjects in the degree program sequence which have prerequisite topics/outcomes that are taught and assessed in the current subject.
- *Program progression* – provides whole-of-program visualisations showing the percentage of planned topic/outcome coverage and planned mastery levels. These are represented via a collection of charts which allow drill-down from high-level knowledge areas to specific topics and outcomes, and to the subjects, assessments and questions where they were assessed. The design and effectiveness of these reports has been presented elsewhere (Gluga, Kay & Lister 2012).

The final tab on the top of Figure 2, *Student Grades*, allows the subject lecturer to view the learning design in terms of the topics/outcomes mapped to exam/assessment questions. It additionally allows the lecturer to import a CSV file containing itemised student grades for each assessment task. Once the grades are imported, the lecturer can generate charts such as the one in Figure 3. These are discussed in the following section as part of our evaluation.

5 Evaluation

To evaluate the conceptual model presented earlier, we initially mapped the final exams from seven core subjects from a computer science degree program offered by an elite Australian ("Go8") university. The questions from each final exam were mapped to the relevant topics/outcomes from the ACM/IEEE CS2013 Strawman draft, at appropriate levels of mastery using Bloom's Taxonomy. This enabled us to generate the Program Progression reports mentioned in the previ-

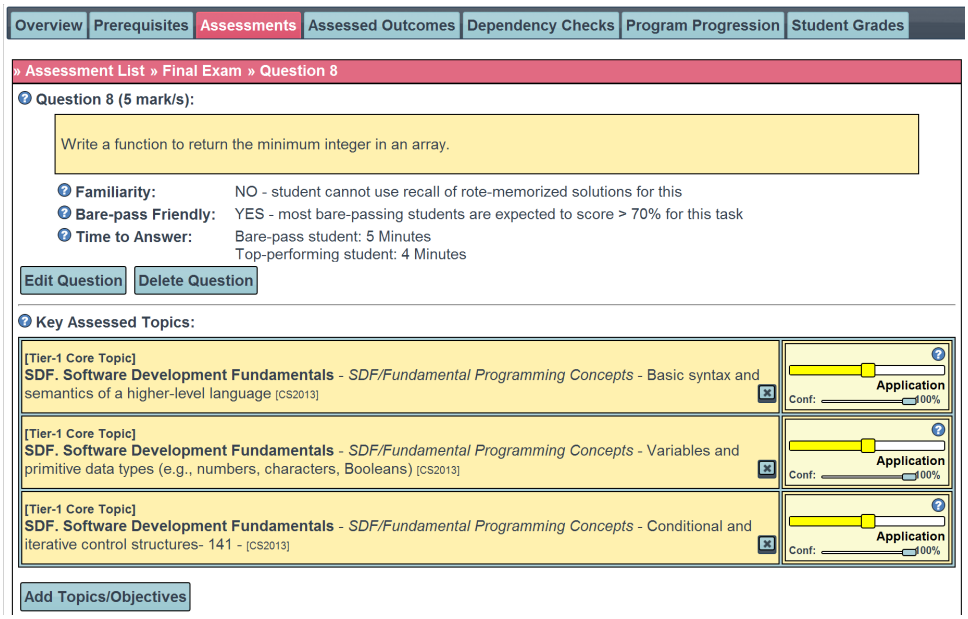

Figure 2: Interface for mapping topics/outcomes and mastery levels to a fictitious question

ous section. This evaluation was described in detail elsewhere (Gluga, Kay & Lister 2012).

The objective of this paper is to evaluate the conceptual model for comparing the assessed curriculum expectations for bare-passing vs. top-performing students against the actual learning achieved, as demonstrated by student grades for each itemised assessment question. To do this, we used the system to code final exams from four programming subjects, each from a different Australian university. The questions from each exam were classified by authors of this paper using the described meta-tags, and validated by an academic involved in the teaching or design of each subject. For one of these four subjects, we also coded the additional three assessments in the subject (two practical tests and one take-home assignment in addition to the final exam). This allowed us to create models of expected learning that took account of the whole of the assessments for the subject.

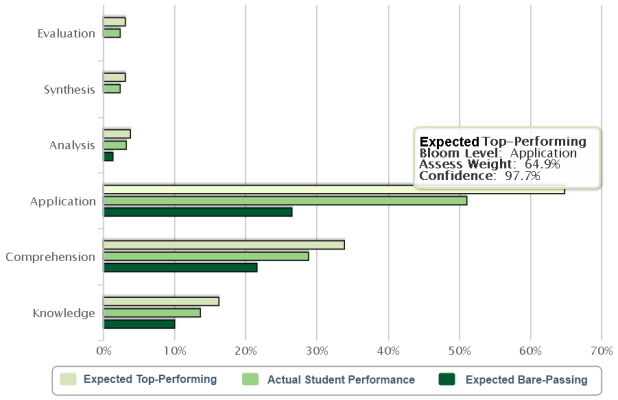

Figure 3: Intended vs. actual performance of top 5% of class in terms of Bloom levels for one subject

For each of these four subjects, we then imported itemised student grades for the four final exams, and also for the three additional assessments for the subject for which we had that information. The numbers of student records imported for the four subjects were 148, 160, 225 and 99. With this data, each lecturer is able to select a subset of their students' grades (for example, the top 10% of students, the bottom 12 students, the 15 students who scored lowest of those who passed) and the system will generate charts such as the one shown in Figure 3. Along the y-axis are the six Bloom levels, with Knowledge at the bottom and Evaluation at the top.

For each Bloom level, the chart shows three bar values: the top bar is the *expected top-performing* student performance; the middle bar is the *actual student performance* of the selected *subset* of the imported grades; and the bottom bar is the *expected bare-passing* student performance. The x-axis represents the overall subject assessment weight associated with each of the Bloom levels. So, for the example in Figure 3, 65% of the total assessment weight for this subject is mapped at the Application Bloom level for top-performing students (top bar), while the bare-passing students (bottom bar) are expected to achieve 27% of these marks. The selected subset of students (middle bar – in this case the top 5% of the class) achieved 52%. Likewise, the subject had 3% of its assessment weight at the Evaluation level for top-performing students, while bare-passing students were not expected to gain any of those marks. Similarly, the Synthesis and Analysis levels had 3% and 4% of assessment weight for top-performing students and bare-passing students were expected to gain up to 1% of the marks at the Analysis level. It is not our intention to judge whether this mapping of assessment weightings to Bloom levels is appropriate. That is a decision that each university must make for itself. It is merely our intention to make decisions of this sort transparent, so that they can be more readily discussed within an institution.

Hovering the cursor over each bar in the chart brings up a tooltip as seen in Figure 3, which indicates the type of student being modelled (Expected Top-Performing), the Bloom mastery level (Application), the percentage of assessment weight associated with that mastery level (64.9%) and finally the confidence or reliability score for this value (97.7%).

This confidence score is based on the reliability of each assessment and the confidence scores for the topic/outcome mappings as described in the previous section. This provides an indication of the level of accuracy of each of the values. So in the given example, the Application level has an overall reliability score of 97.7%, meaning the classifiers were very confident when mapping questions to the Application level. The Knowledge level, however, had a confidence score of 69%. The final exam in this evaluation was given a re-

liability score of 100%, as it was closed-book and taken under strict supervision. This implies that many of the questions which were mapped at the Knowledge level had low confidence scores associated with them. This is perhaps because the classifier was unsure if the students would use rote learning to answer the question, or reason about the solution using higher cognitive skills. Most of the Application level questions, however, had a 100% confidence rating associated with them.

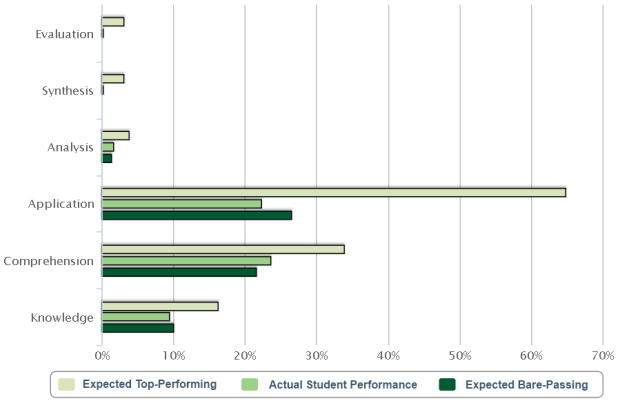

Figure 4: Intended vs. actual performance of bottom 5% of bare-passing students in terms of Bloom levels

The system allows the user to generate this chart for any subset of actual student grades. That is, after importing the student marks, the user may select one or more students to be included in the computation for the middle bar. If only one student is selected, the chart represents the demonstrated curriculum for that individual learner. If, say, five students are selected, the chart shows the demonstrated curriculum as an average across this subgroup. This allows flexible comparison of the expected performance with, say, the actual performance of top students, the actual performance of bare-passing students, the actual performance of the class as a whole, etc.

The middle bar in the chart in Figure 3 shows the actual performance of the top 5% of the class (i.e. the top 12 students, who scored between 82% and 90%). This reveals that the actual top-performing students in this example scored between the expected top and expected bare-passing levels, but closer to the former. Compare this to the chart in Figure 4, which shows the actual performance for the bottom 5% of bare-passing students for the same cohort (that is, the 12 students who scored the lowest marks of 50% or more, which ranged between 50 and 52). This reveals that the actual bare-passing students are scoring marks below the expected bare-passing marks for the Application and Knowledge levels. We could similarly regenerate this chart for the whole class, for a single student, or for any other subset of interest.

The charts in Figures 3 and 4 represent the overall assessment distribution in terms of Bloom levels. A different visualisation allows the user to see the overall assessment distribution in terms of the mapped syllabus topics/outcomes, as shown in Figure 5. The CS2013 topics/outcomes that were mapped to exam questions for this subject are shown along the y-axis. The x-axis shows the overall assessment weight associated with each topic/outcome. The three bars in each series have the same meaning as in the previous two charts, that is, expected top-performing students as the top bar, actual student performance as the middle bar, and expected bare-passing students as the bottom bar. The image in Figure 5 is cropped to show only the bottom five topic/outcome mappings. The actual chart in the system is scrollable, and in the case of this subject's final exam it shows 43 such mappings. The chart in Figure 5 shows the actual performance of the bottom 12 bare-passing students. For the five topics/outcomes shown, the actual bare-passing performance is very close to the mapped intended bare-passing performance.

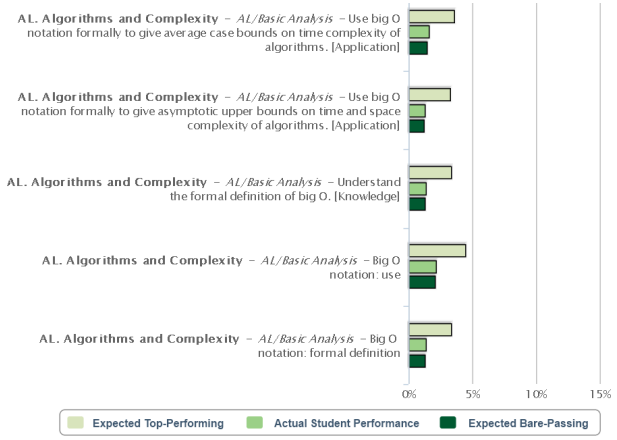

Figure 5: Intended vs. actual performance of bottom 5% of bare-passing students in terms of topics/outcomes (partial)

The charts also support drill-down functionality. Clicking on any of the three bars in the Application series in Figure 3, for example, will bring up a new chart that identifies the assessment weights and reliabilities associated with all of the topics/outcomes that were assessed at the Application level. Further clicking on any bar in the new chart will bring up a dialog showing all of the exam questions that contributed to the clicked-on topic. Likewise clicking on a topic/outcome bar in Figure 5 will bring up a new chart that provides a breakdown of the Bloom levels at which that topic/outcome was assessed, and a further click will bring up the exam questions contributing to those mappings.

5.1 Participant Feedback Results

For each of the four subjects mapped into the system, an academic involved in the teaching or design of that subject was asked first to validate the meta-tags in each question classification, and then to use the charting visualisations described above to compare the assessment design against demonstrated student performance across different groups of students. After doing so, the academics were asked to complete a questionnaire with Likert-scale responses and open-answer feedback commenting on the perceived usefulness of the approach and system implementation. The hypotheses for this evaluation were:

1. Differentiating between the expected outcomes of bare-passing and top-performing students is useful.
2. Comparing expected bare-passing/top-performing outcomes against actual bare-passing/top-performing student outcomes is useful.
3. Visualising the assessment distribution of the subject in terms of mastery levels is useful.
4. Visualising the assessment distribution of the subject in terms of syllabus topics and outcomes is useful.

5. Expressing reliability of classifications is useful.
6. The system interfaces for classifying and visualising information are effective.
7. Academics would consider using the system to model their own subjects and assessments if it were available to them.

The term 'useful' in this context is used to capture whether or not participants perceived value in the approach and in the rich reporting interfaces that allowed them to compare expected learning vs. actual learning in their assessments. These were tested using a series of Likert-scale questions that mapped to each of the hypotheses (at least two questions mapped to each hypothesis, with some questions mapping to multiple hypotheses). The scale ranged from 1 (Strongly Disagree) to 5 (Strongly Agree). The average scores for the seven hypotheses were all in agreement (H1=4.25, H2=4.25, H3=4.13, H4=4, H5=3.75, H6=4.63, H7=4.25).

Open-ended feedback by the participating academics was also of high interest. One participant commented in relation to H1 that "I suspect this is something that I always have in the back of my mind when setting assessments...So what this has done is bring these thoughts to the fore and make them explicit on a question-by-question basis". Increasing the transparency of these assessment design decisions, so that they may be shared across subjects, is an important outcome. One participant commented that "I can see that this would be useful in terms of syllabus design, but once the course is designed and implemented I think the usefulness diminishes". This may be true to some extent, namely that the approach would be most useful in the initial design of a new subject or new degree program. However, subjects, subject lectures, degree program enrolment rules and even curriculum recommendations do often change. Having the original design decisions explicitly captured will enable more informed restructuring of teaching and learning activities at these points. For example, suppose an academic designs a new subject, including all assessments, so that it aligns with a set of recommended learning goals. What happens when this academic leaves and is no longer responsible for this subject? How does a subsequent lecturer know the implicit reasoning behind the assessment design?

While the participants indicated overall satisfaction with the effectiveness of the system interfaces, some were concerned that the initial data entry may be somewhat time-consuming. Additionally, some participants found that the mapping of questions to the CS2013 topics and outcomes was not always obvious. In particular, a number of exam questions were identified where the primary assessed concepts were not found in the CS2013 specification (e.g. variable scope and static variables). However, overall the participants were satisfied with the use of mastery levels to differentiate between the performances of different student groups. One participant stated "A very useful tool. It has suggested, for instance, a broad difference in the Bloom level that students reach in different bands: below a Credit (65%) for instance, the Application level average performance dips below the performance for Comprehension-level questions. Interesting stuff which might give a good perspective to academics who are hoping to define clearly what it means to be a 'Credit level' student vs a 'Passing' student."

6 Discussion

The evaluation presented in this paper used student marks from four subjects, each from a different institution, to validate the conceptual design. For one of the four subjects we were able to import marks for all of the assessments, not just for the final exam. This provided a more realistic picture of the learning expectations vs. actual performance across that entire subject. When considering only final exams, the generated reports showed very small assessment weight at the higher Bloom levels (Synthesis and Evaluation). This is to be expected, as testing for competence at these higher levels is typically more appropriate in larger design-oriented tasks such as take-home assignments. This was reflected in the subject for which we had student results for all assessments: a large portion of a 20% take-home assignment was classified at the Bloom Synthesis level. The charts for this report thus contrasted with those from the three subjects with only final exams, which had very little emphasis on these higher levels.

However, take-home assignments may have lower reliability scores, depending on the classifier's judgement. This would thus reduce the confidence scores for these upper Bloom levels, as compared to the stronger reliability for the lower levels assessed in final exams. Appropriate ways in which to interpret these confidence and reliability scores need to be explored. If a topic/outcome has an overall confidence value of less than 50%, what can we claim about the knowledge of the student at the end of the subject? Perhaps the reliability scores are overly pessimistic and need to be raised? Perhaps some of the confidence values associated with each question mapping are too low and need to be revised? Perhaps the subject relies too heavily on less trusted assessment techniques and thus cannot support strong claims about the learning outcomes of the passing student? In any case, the conceptual design and implementation allows the academic to explicitly document and capture these concerns, providing the opportunity to iteratively refine the learning design so as to raise the mastery levels and their confidence values to appropriate levels.

The conceptual design also supports the generation of similar reports and visualisations across a sequence of subjects, or across a full degree program. This may provide very valuable information for degree program quality assurance and accreditation purposes, and for communicating with employers or other stakeholders a more precise picture of graduate capabilities. The main difficulty in doing this is collecting the itemised fine-grained student marks for each individual question in each subject assessment. This may require a change to current assessment processes in some institutions, where typically marks are stored only at a coarse level. For example, only a single mark for a quiz or exam is recorded in the student gradebook system, and after the student completes a subject, these itemised marks are often lost and all that remains is an overall subject mark for each student, which does not provide sufficient information for such analysis.

The validation presented in this paper maps topics and outcomes from the CS2013 Strawman curriculum guideline, which is not formally accredited in Australian computer science degree programs. Perhaps an institution may be more interested in mapping assessment tasks against the ACS Core Body of Knowledge, or the Skills Framework for the Information Age (SFIA 2012) as proposed in the new ACS accreditation guidelines. The skills, attributes and topics listed in these documents are significantly higher-level, so instead of mapping to each individual exam question, it may be more appropriate to classify only at the assessment level, or even the subject level as a whole. Such higher-level attributes and skills are discussed at length in Gluga, Lever & Kay (2012).

The model presented is agnostic of any specific syllabus or body-of-knowledge statement, so it could

instead be used with any internally defined taxonomy of topics or concepts that an institution, department or group of academics decides on as important. Additionally, the model is agnostic of the method by which mastery levels are classified. We have used Bloom's Taxonomy, as it has received significant attention in computing education, but other classification schemes such as neo-Piagetian cognitive development (Lister 2011), the SOLO Taxonomy (Sheard et al. 2008), or any internally defined scheme may work equally well.

The evaluation and discussion thus far have focused on a single offering of each subject. That is, the final exams and student results were from a particular offering of each of the four classified subjects. The charts and reports shown here are thus only a snapshot view of a subject or collection of subjects at a particular point in time. The envisaged use of the system is to model lecturer expectations from a subject offering, then to compare these expectations to actual student performance at the end of the offering, and to take any necessary corrective action in the teaching and learning design or assessment design for the next offering. That is, the conceptual model is intended to be used as a tool for iterative improvement of courses and programs. The snapshot aspect of the data might appear somewhat restrictive, in that each new offering of a course would entail new data. However, it is our experience that while assessment items generally change from one offering to the next, what they assess and how they assess it remain fairly constant; therefore all that is required for a new offering is to check the data for the previous offering and adjust it appropriately.

The primary concern in using the tool for this purpose and in this fashion, as expressed by some of our participants, is the perceived effort required in performing the fine-grained classifications. However, as reported in Gluga, Kay, Lister & Lever (2012), the time taken for mapping a full exam paper is between one to two hours, or slightly more depending on the granularity of questions. Mapping additional assessments from the subject may thus take a further hour or two. An entire subject can thus be reasonably classified by the lecturer of that subject within a single sitting. This would enable very rich long-term models of the curriculum with a modest time investment from each of the 24 or more subject lecturers.

7 Conclusion

To design an effective computer science degree program, subject lecturers need to have a clear understanding of the learning standards that they are to teach and assess, and the capabilities of their students at the beginning and end of each subject. That is, lecturers must know what syllabus topics and outcomes the students have previously learnt, and what mastery level they have attained, in order to design effective teaching, learning and assessment activities that integrate appropriately into the overall degree program sequence. Additionally, lecturers need to be aware of the differences in capabilities between the bare-passing students and top-performing students, to help ensure that neither group is neglected. Likewise, subject lecturers must be able to communicate this knowledge amongst themselves as students progress through the many subjects of a degree. They must additionally be able to support this knowledge with evidence based on actual student grades, such that any unmet expectations can be addressed in future revisions of the curriculum.

To achieve these goals, we have presented a conceptual model that supports the description of subject assessment questions in terms of syllabus topics or outcomes, such as the CS2013, and also in terms of mastery levels, such as Bloom's Taxonomy. The model additionally supports the importing of student marks to represent the actual Demonstrated Curriculum, which we believe to be important for iterative teaching refinement. A third component of the model is the capture of reliability scores for each assessment task and confidence ratings in each question classification. These are useful for representing the accuracy and reliability of the generated curriculum models, on which important decisions may be based.

We have validated the conceptual model by creating a web-based implementation that enables users to enter all the subject assessment data and to effectively classify each individual question. The system was used to model the Assessed Curriculum based on the final exams of seven core programming subjects from a real computer science degree program. To test the effectiveness of comparing the expected learning outcomes with actual student performance, we imported itemised final exam marks from 632 students across four programming subjects from different institutions. The system was used to aggregate these grades against the question classifications and present a series of charts that allow visualisation of the data from multiple perspectives. Additionally, for one subject we were able to import student marks for all remaining assessments, enabling us to generate realistic reports as to the learning design of that subject as a whole, and to compare that against the Demonstrated Curriculum.

Academics involved in the teaching or delivery of each of the four subjects validated the question classifications and experimented in using the charting visualisations to explore how closely their expectations of bare-passing vs. top-performing students matched the actual student performance at different band levels. Overall, the academics expressed positive interest in using a similar system to document and visualise their subjects and assessments.

The main contributions of this paper are the conceptual model for capturing the learning design and expectations, for comparing these against demonstrated student performance, and for also capturing the reliability of the generated models. The system is freely available to trial online at http://progoss.com.

Acknowledgements

This work was supported by a grant from the Australian Government Office for Learning and Teaching.

References

ABET (2011), 'ABET Computing Accreditation Commission, Criteria for Accrediting Computing Programs', http://www.abet.org/cac-current-criteria/.

Abunawass, A., Lloyd, W. & Rudolph, E. (2004), COMPASS: a CS program assessment project, *in* 'ACM SIGCSE Bulletin', Vol. 36, ACM, pp. 127–131.

ACM/IEEE (2008), 'Association for Computing Machinery and the IEEE Computer Society, Computer Science Curriculum 2008 (CS2008)', http://www.acm.org/education/curricula/ComputerScience2008.pdf.

ACM/IEEE (2013), 'Association for Computing Machinery and the IEEE Computer Society, Computer Science Curricula 2013 (CS2013)', http://ai.stanford.edu/users/sahami/CS2013/.

ALTC (2010), 'Engineering and ICT: Learning and Teaching Academic Standards Statement'. **URL:** *http://www.olt.gov.au/resource-engineering-ict-ltas-statement-altc-2010*

Barrie, S., Hughes, C. & Smith, C. (2009), 'The Na-

tional Graduate Attributes Project: integration and assessment of graduate attributes in curriculum'.

Bloom, B. S., Engelhart, M. B., Furst, E. J., Hill, W. H. & Krathwohl, D. R. (1956), *Taxonomy of educational objectives. The classification of educational goals. Handbook 1: Cognitive domain*, Longmans Green.

Britton, M., Letassy, N., Medina, M. & Er, N. (2008), 'A curriculum review and mapping process supported by an electronic database system', *American journal of pharmaceutical education* **72**(5).

Buck, D. & Stucki, D. J. (2001), JKarelRobot: A case study in supporting levels of cognitive development in the computer science curriculum, *in* 'Proceedings of the Thirty-second SIGCSE Technical Symposium on Computer Science Education', SIGCSE '01, ACM, New York, NY, USA, pp. 16–20.

Burgess, G. A. (2005), 'Introduction to programming: Blooming in America', *J. Comput. Sci. Coll.* **21**(1), 19–28.

English, F. (1978), 'Quality control in curriculum development'.

English, F. (1988), 'Curriculum auditing'.

Gluga, R., Kay, J. & Lever, T. (2010), Modeling long term learning of generic skills, *in* V. Aleven, J. Kay & J. Mostow, eds, 'Intelligent Tutoring Systems', Vol. 6094, Springer, pp. 85–94.

Gluga, R., Kay, J. & Lister, R. (2012), ProGoSs: Mastering the Curriculum, *in* M. Sharma & A. Yeung, eds, 'Australian Conference on Science and Mathematics Education (ACSME2012)', 18th Annual UniServe Science Conference, UniServe Science, The University of Sydney, NSW 2006, Australia, pp. 92–98.

Gluga, R., Kay, J., Lister, R., Kleitman, S. & Lever, T. (2012), Coming to terms with Bloom: An online tutorial for teachers of programming fundamentals, *in* M. de Raadt & A. Carbone, eds, 'Australasian Computing Education Conference (ACE2012)', Vol. 123 of *CRPIT*, ACS, Melbourne, Australia, pp. 147–156.

Gluga, R., Kay, J., Lister, R. & Lever, T. (2012), A unified model for embedding learning standards into university curricula for effective accreditation and quality assurance, *in* 'Australasian Association for Engineering Education (AAEE2012)', Vol. [to appear], Melbourne, Australia.

Gluga, R., Lever, T. & Kay, J. (2012), 'Foundations for modelling university curricula in terms of multiple learning goal sets', *IEEE Transactions on Learning Technologies* p. to appear.

Gregor, S., von Konsky, B. & Wilson, D. (2008), 'The ICT profession and the ICT body of knowledge (vers. 5.0)', http://www.acs.org.au/attachments/ACSCBOKWorkingPaper2008.pdf.

Harden, R. (2001), 'Amee guide no. 21: Curriculum mapping: a tool for transparent and authentic teaching and learning', *Medical Teacher* **23**(2), 123–137.

Jacobs, H. (1989), *Interdisciplinary curriculum: Design and implementation*, ERIC.

Jacobs, H. (1991), 'Planning for curriculum integration', *Educational Leadership* **49**, n2.

Jacobs, H. (1997), *Mapping the Big Picture. Integrating Curriculum & Assessment K-12*, ERIC.

Jacobs, H. (2010), *Curriculum 21: Essential education for a changing world*, Association for Supervision and Curriculum Development.

Lister, R. (2011), Concrete and Other Neo-Piagetian Forms of Reasoning in the Novice Programmer, *in* J. Hamer & M. de Raadt, eds, 'Australasian Computing Education Conference (ACE 2011)', Vol. 114 of *CRPIT*, ACS, Perth, Australia, pp. 9–18.

Lister, R. (2012), 'The CC2013 Strawman and Bloom's taxonomy', *ACM Inroads* **3**(2), 12–13.

Lister, R., Corney, M., Curran, J., D'Souza, D., Fidge, C., Gluga, R., Hamilton, M., Harland, J., Hogan, J., Kay, J. et al. (2012), Toward a shared understanding of competency in programming: an invitation to the babelnot project, *in* 'Proceedings of the 14th Australasian Computing Education Conference (ACE2012)', Vol. 123, Australian Computer Society.

Lister, R. & Leaney, J. (2003*a*), First Year Programming: Let All the Flowers Bloom, *in* 'ACE '03: Proceedings of the Fifth Australasian Computing Education Conference', Australian Computer Society, Inc., Darlinghurst, Australia, pp. 221–230.

Lister, R. & Leaney, J. (2003*b*), 'Introductory programming, criterion-referencing, and Bloom', *SIGCSE Bull.* **35**(1), 143–147.

McGettrick, A. (2005), 'Grand challenges in computing: Education–a summary', *The Computer Journal* **48**(1), 42–48.

Oliver, D., Dobele, T., Greber, M. & Roberts, T. (2004), This course has a Bloom rating of 3.9, *in* 'Proceedings of the Sixth Australasian Computing Education Conference – Volume 30', ACE '04, Australian Computer Society, Inc., Darlinghurst, Australia, pp. 227–231.

Reynolds, C. & Fox, C. (1996), Requirements for a computer science curriculum emphasizing information technology: subject area curriculum issues, Vol. 28, ACM, pp. 247–251.

Sadler, D. (2009), 'Moderation, grading and calibration'.

SFIA (2012), 'Skills Framework for the Information Age: How SFIA Works'.
URL: *http://www.sfia.org.uk*

Sheard, J., Carbone, A., Lister, R., Simon, B., Thompson, E. & Whalley, J. L. (2008), 'Going SOLO to assess novice programmers', *SIGCSE Bull.* **40**, 209–213.

Simon, Sheard, J., Carbone, A., Chinn, D., Laakso, M.-J., Clear, T., de Raadt, M., D'Souza, D., Lister, R., Philpott, A., Skene, J. & Warburton, G. (2012), Introductory programming: examining the exams, *in* M. de Raadt & A. Carbone, eds, 'Australasian Computing Education Conference (ACE2012)', Vol. 123 of *CRPIT*, ACS, Melbourne, Australia, pp. 61–70.

Starr, C. W., Manaris, B. & Stalvey, R. H. (2008), 'Bloom's taxonomy revisited: specifying assessable learning objectives in computer science', *SIGCSE Bull.* **40**(1), 261–265.

Thompson, E., Luxton-Reilly, A., Whalley, J. L., Hu, M. & Robbins, P. (2008), Bloom's taxonomy for CS assessment, *in* 'Proceedings of the Tenth Australasian Computing Education Conference – Volume 78', ACE '08, Australian Computer Society, Inc., Darlinghurst, Australia, pp. 155–161.

Whalley, J. L., Lister, R., Thompson, E., Clear, T., Robbins, P., Kumar, P. K. A. & Prasad, C. (2006), An Australasian study of reading and comprehension skills in novice programmers, using the Bloom and SOLO taxonomies, *in* 'Proceedings of the 8th Australasian Computing Education Conference – Volume 52', ACE '06, Australian Computer Society, Inc., Darlinghurst, Australia, pp. 243–252.

Wigal, C. (2005), Managing and aligning assessment knowledge, *in* 'Frontiers in Education, 2005. FIE'05. Proceedings 35th Annual Conference', IEEE, pp. T3C–13.

Willett, T. (2008), 'Current status of curriculum mapping in Canada and the UK', *Medical education* **42**(8), 786–793.